

If Emulation of Another System Is Necessary, Ensure that It Is as Correct and Complete as Possible

William L. Fithen, Software Engineering Institute [vita³]

Copyright © 2005 Carnegie Mellon University

2005-10-03

L4 / D/P⁴

Incorrect or incomplete emulation can introduce vulnerability.

Description

In general, an emulation fidelity vulnerability exists when

- a system must emulate another system or device,
- that emulation is incorrect or incomplete, and
- the system uses the emulated state information to make security decisions.

The defect might be some or all of the following:

- Emulation that is too abstract. Many network-based intrusion detection systems passively watch traffic of other systems, trying to guess the state of end nodes in communications with one another based on communication fragments. Packet-based firewalls in certain configurations exhibit this same shortcoming. For both of these examples, complete emulation is not generally possible because many end-node policies that influence their state are not observable in the traffic.
- The emulator is simply wrong (i.e., logic error) and does not emulate the original correctly.
- The emulation is correct, but it does not perform in realtime. That is, it cannot keep up with what it's emulating, resulting in a denial of service.

References

- | | |
|--------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| [Hoglund 04] | Hoglund, Greg & McGraw, Gary. <i>Exploiting Software: How to Break Code</i> . Boston, MA: Addison-Wesley, 2004. |
| [VU#548515] | Finlay, Ian. <i>Vulnerability Note VU#548515: Multiple intrusion detection systems may be circumvented via %u encoding</i> . http://www.kb.cert.org/vuls/id/548515 (2003). |

Carnegie Mellon Copyright

Copyright © Carnegie Mellon University 2005-2010.

This material may be reproduced in its entirety, without modification, and freely distributed in written or electronic form without requesting formal permission. Permission is required for any other use. Requests for permission should be directed to the Software Engineering Institute at permission@sei.cmu.edu¹.

The Build Security In (BSI) portal is sponsored by the U.S. Department of Homeland Security (DHS), National Cyber Security Division. The Software Engineering Institute (SEI) develops and operates BSI. DHS funding supports the publishing of all site content.

NO WARRANTY

3. http://buildsecurityin.us-cert.gov/bsi/about_us/authors/320-BSI.html (Fithen, William L.)

1. <mailto:permission@sei.cmu.edu>

THIS MATERIAL OF CARNEGIE MELLON UNIVERSITY AND ITS SOFTWARE ENGINEERING INSTITUTE IS FURNISHED ON AN "AS-IS" BASIS. CARNEGIE MELLON UNIVERSITY MAKES NO WARRANTIES OF ANY KIND, EITHER EXPRESSED OR IMPLIED, AS TO ANY MATTER INCLUDING, BUT NOT LIMITED TO, WARRANTY OF FITNESS FOR PURPOSE OR MERCHANTABILITY, EXCLUSIVITY, OR RESULTS OBTAINED FROM USE OF THE MATERIAL. CARNEGIE MELLON UNIVERSITY DOES NOT MAKE ANY WARRANTY OF ANY KIND WITH RESPECT TO FREEDOM FROM PATENT, TRADEMARK, OR COPYRIGHT INFRINGEMENT.